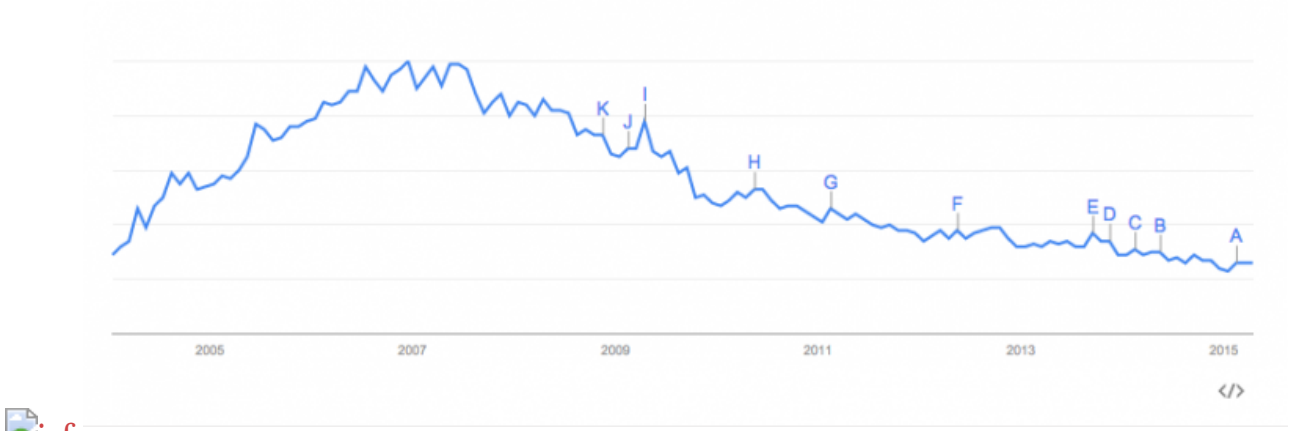
¿Tiene futuro JSF?

21 abril, 2015 por Cecilio Álvarez Caules — 95 comentarios

Todos hemos trabajado en unos u otros proyectos con JSF y su tecnología orientada a controles. Una pregunta que cada día me viene más a la cabeza es: **¿Cual es el futuro de JSF?.** Para muchas personas la respuesta tiende a ser sencilla . **JSF es el estandar de Java EE a nivel de capa de presentación por lo tanto es el futuro.** Sin embargo ya hemos tenido en otras ocasiones standards de la plataforma Java cuyo que han pasado a mejor vida o han tenido que reinventarse de forma significativa. Quizás para poder valorar más cual es el estado de JSF debemos realizar una consulta en Google trends (tendencias) y obtener el siguiente resultado.

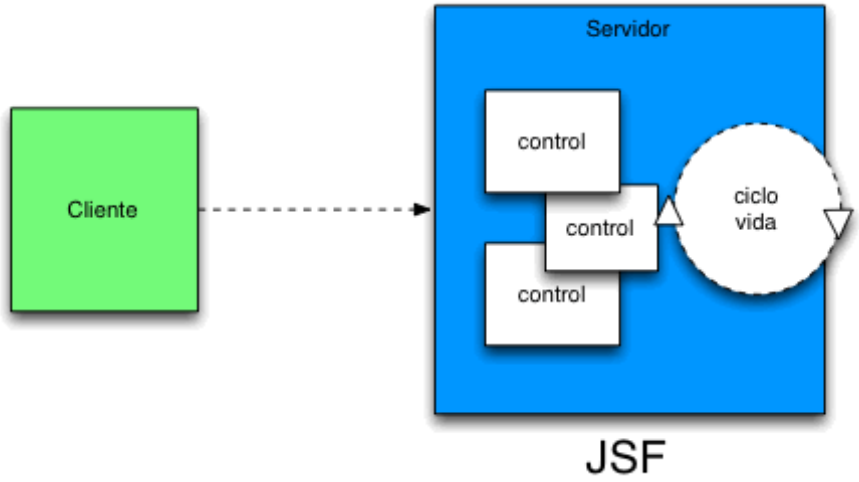


jsf

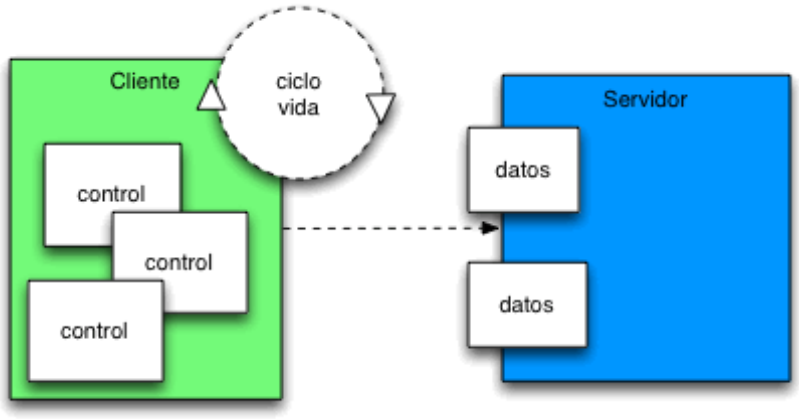
Es evidente que el interés por JSF esta decreciendo dentro de la comunidad. **¿Porqué?** esa es una buena pregunta ya que recordemos que JSF es el standard **y ha evolucionado de una forma bastante sólida e interesante durante los últimos años** . Por decirlo de alguna forma tendríamos que esperarnos una gráfica completamente al revés. **¿Qué es lo que ha pasado?.**

JSF y Servidor

JSF siempre ha gestionado su ciclo de vida a través del servidor construyendo los distintos tipos de controles que necesitamos usar. Dejando al cliente unas responsabilidades bastante reducidas

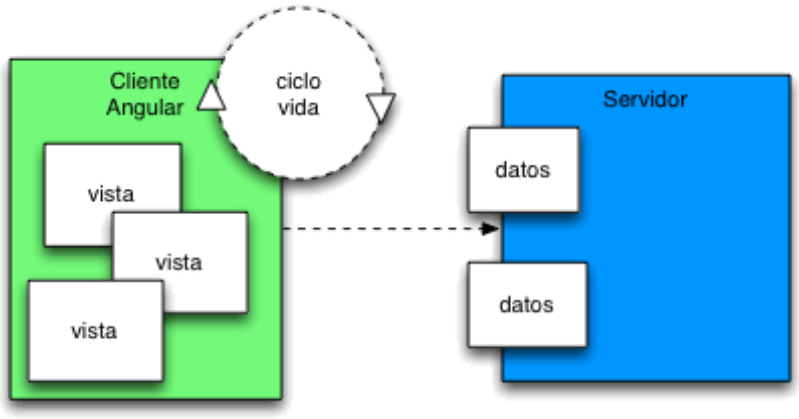


Hoy por hoy muchas aplicaciones han optado por delegar **una gran parte de las responsabilidades al cliente** . Ejemplos claros son las plataformas móviles con HTML5 ,Arquitecturas SPA y frameworks como JQuery Mobile que usan simplemente el servidor para nutrirse de datos pero **que todos los controles vienen definidos directamente en el cliente.** Ante este tipo de situaciones JSF pierde parte de su interés.



Framework MVC cliente (Angular, Backbone etc)

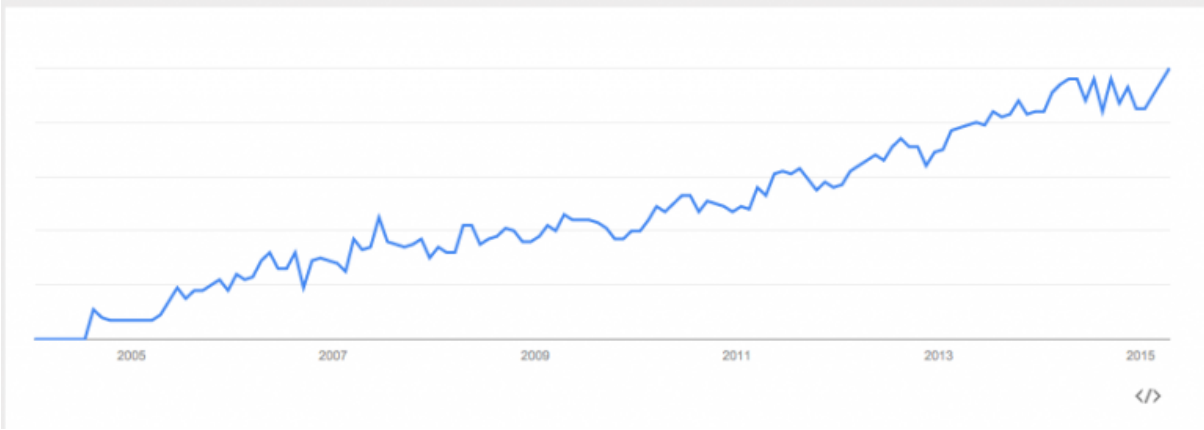
Aparte de lo que acabamos de comentar lo**s nuevos Frameworks MVC JavaScript** que se ejecutan en cliente se apoyan en una filosofía similar a las aplicaciones móviles a la hora de asumir responsabilidades de cliente . **Con la ventaja ademas de que están orientados a desarrollar aplicaciones de gran tamaño.**



Así pues entre los frameworks móviles y los Frameworks MVC el pastel se tiene que repartir entre mucha mas gente y **JSF pierde poco a poco cuota de mercado**.

Spring MVC

Spring MVC hace tiempo que compite con JSF pero su enfoque no esta orientado a controles sino que se basa mas en un enfoque **MVC puro** basado en controladores. La ventaja de este tipo de soluciones es que su integración con las nuevas arquitecturas móviles y JavaScript MVC **es más directa.** De hecho si revisamos google trends podremos ver que cada día tiene mejor acogida por parte de la comunidad.



JSR 371 MVC (La solución)

Los standards se están haciendo eco de las necesidades de la comunidad y esperemos **que la nueva especificación de JSR 371 que se denomina MVC nos aporte un modelo MVC dentro del standard** que permita trabajar de forma integrada con todos los frameworks cliente que existen.

¿Abandonar JSF?

¿Debemos abandonar JSF? .Esta es una pregunta **que mucha gente se esta haciendo**. No creo que tengamos **que ser tan drásticos,** JSF sigue siendo una tecnología que podemos usar **y que tiene su hueco** . Quizás hoy por hoy encaje mas en entornos intranet que internet. Pero lo que cada día esta más claro es que otras arquitecturas **están apareciendo en el horizonte y tendremos que abordarlas** de una forma u otra. Para que nuestras soluciones mantengan su flexibilidad.

Otros artículos relacionados: JSF y HTML5 , Spring MVC , JSF Ajax

Descargar PDF

Archivado en: Java EE, Java Web, JSF, mas leido, Spring MVC
Etiquetado como: JavaeeTips

Comentarios

Ivan dice
30 octubre, 2017 en 18:51

Me parece interesante la descripcion que se realiza, sin embargo las aplicaciones empresariales que por lo menos me ha tocado realizar en el ultimo tiempo utilizan los framework como Boostrap o Angular solo en presentacion, la logica

generalmente es desarrollada para que se ejecute en el server, el concepto de RIA para aplicaciones HTML5 no existe completamente, primero por que funciona mal por las distintas implementaciones de los browsers, lo unico que genera intentar realizar ese trabajo es tener programas llenos de if si el browser es mozilla-edge-chrome, adicionalmente uno siempre intenta tener un sistema que sea facil de mantener, operar, modificar y todas las nuevas tecnologias que ofrecen los framework como Spring son lentas y expurias (si se ha dado el trabajo de leer el codigo interno). Como por ejemplo al utilizar Boostrap el trafico que genera es casi incomprensible, ver todo lo que se traslada al cliente. Por lo cual a mi opinion si bien el concepto de MVC es necesario y absolutamente correcto, creo que falta por optimizar y modularizar. ADF es una excelente opcion, sobre todo para los que trabajan con JSF y quieren migrar a algo con muy poco esfuerzo.

Responder

Cecilio Álvarez Caules dice
4 noviembre, 2017 en 15:28

JSF es una opción más pero hoy por hoy cada día los frameworks MVC que publican información en JSON tienen mucho futuro

Responder

Paul dice
26 julio, 2017 en 22:02

Métanse a ADF ¿por qué pierden el tiempo?, ADF es JSF, brinda mayores satisfacciones que PrimeFaces que también es JSF.

Responder

Cecilio Álvarez Caules dice
27 julio, 2017 en 13:44

Gracias por el aporte ☺

Responder

Miguel Nolasco dice
23 octubre, 2017 en 1:01

Buena opción, si se deciden por usar JSF mejor es usar ADF de Oracle y es libre. Puede ser implementado en Glassfish, JBoss, Wildfly.

Responder

Jose dice
20 marzo, 2018 en 19:49

Nunca habia escuchado de ADF.
JSF, Spring, Hibernate los tiene NetBeans como frameworks.

Podrias explicar un poco mas, donde obtengo informacion, gracias.

Responder

Cecilio Álvarez Caules dice
2 abril, 2018 en 19:54

Uff no sabría decirte igual aquí

Responder

madoc dice
26 julio, 2018 en 8:35

PrimeFaces no es JSF , es una extension de JSF.

Responder

Alfredo dice
24 junio, 2017 en 2:23

Muy buen post. Estoy de acuerdo en lo que has publicado, yo agregaría un punto adicional: “Licenciamiento”. Ya vimos la guerra que Oracle le ha declarado a Google, y creo que esto también debe ser un punto importante si no quieres gastar dinero en el futuro con abogados.

Responder

Jose dice
27 marzo, 2017 en 5:16

Hola

Escribo desde Chile, quisiera saber si aún está vigente JSF en el 2017 ya que estoy por desarrollar un proyecto para una Clinica Intranet, me preocupa el impacto que puede tener el sistema en el futuro o que pueda tener tropiezos o problemas durante el desarrollo del proyecto.

Muchas gracias.

Responder

Cecilio Álvarez Caules dice
27 marzo, 2017 en 7:10

Es parte de los standares y sigue totalmente vigente. De hecho es una tecnología que tendrá futuro ya que la soporta Oracle. Otra cosa es que existan otras opciones de cliente más interesantes

Responder

Lincoln dice
10 enero, 2017 en 11:09

Cecelio, yo soy desarollador java en Brasil.
Los parabenos el articulo!

Responder

Cecilio Álvarez Caules dice
10 enero, 2017 en 12:26

Buenas ☺ , gracias /thanks

Responder

elenaelena dice

20 diciembre, 2016 en 21:44

Hola, creo que la tecnología jsf se mantendrá pero irá “migrando” la lógica de cliente hacia el lado del cliente.
Me explico?. Hoy por hoy toda la iteraccion y eventos se produce realmente en el browser pero se ejecuta en el server.
Los bakend beans pasaran a ser una especie de “restservices” y la vista hablara con ellos de forma transparente y nosotros ni nos daremos cuenta porque se seguira programando como antes.
Incluso me atrevería a decir que esta funcionalidad podría ser configurable.

Responder

Cecilio Álvarez Caules dice
21 diciembre, 2016 en 10:33

El proyecto de primefaces y angular tiene un aire así

Responder

Janotao dice
8 diciembre, 2016 en 22:51

Muy buen post, sin duda java al ser compilado lo hace mas rapido que los demas lenguajes interpretados.

Y en cuanto a las estadisticas en google trends, puede deberse a que hay una gran cantidad de desarrollo pequeños (paginas web, blogs, paginas personales) con poco recursos para pagar un hosting en java, o servidor dedicado, etc. y usan tecnologias de menor costo, integrados con jquery, angular, boostraps.

Algo asi, usar JSF para mi pagina personal, para un blog, sin duda noo

Entonces JSF lo dedicas mas que todo a aplicaciones empresariales, intranets,

Responder

Cecilio Álvarez Caules dice
9 diciembre, 2016 en 9:12

Gracias , cada tecnología tiene su hueco ☺

Responder

madopiuter dice
26 julio, 2018 en 8:36

Java es semicompilado. Saludos

Responder

Esteban dice
7 diciembre, 2016 en 6:36

Buenisimo el post, en este momento estoy desarrollando una web app para ofrecer un servicio de estadisticas, de momento llevo gran parte pero sin usar frmaworks, he estado realizando el mvc puro a mano, los servlets y el modelo todo a mano, no he tenido problemas, que diferencia tendria en usar spring mvc ya que leo que es puro mvc?, y por cierto seria muy trabajoso migrar mi trabajo a spring o es sencillo? o mejor sigo a mano con el mvc y no es problema?

Gracias si me pueden solventar estas dudas, saludos.

Responder

Cecilio Álvarez Caules dice
7 diciembre, 2016 en 19:30

En principio sencillo sencillo no te será , pero pienso que que si tienes servicios y vistas tipo JSTL es algo abordable. Spring te facilitará las cosas y te aportará muchas cosas adicionales que te serán útiles en el futuro

Responder

Paul dice
18 noviembre, 2016 en 0:55

ADF es JSF, arrastras lo ya programado (tablas, etc) y lo que ya se supone está enlazado a base de datos, y listo!, y por dentro tiene ajax, etc...o sea, tiene bastante futuro. ¿por qué no picas código? para qué, si donde tienes que invertir tiempo es en diseñar la database, luego, la herramienta, Oracle JDeveloper, te permite utilizar este RAD. Cualquier cambio, automáticamente se actualiza. Mejora la productividad. Sin ADF, picas código, repites el código, no pones tiempo en la database,

demuestras que picas código, el resultado, falta la hoja de estilo, aplicas html5 para border redondeados, sorpresa! no funciona en todos lso exploradores, sigues picando código para arreglarlo, sorpresa! el analista pensó en cambiar una tabla, picas código para acomodar este DAO, después de meses deciden cambiar la apariencia, sorpresa! a cambiar los estilos, y así picando código para algo que puedes resolver simplemente con drag & drop y con ADF,

Responder

Cecilio Álvarez Caules dice
18 noviembre, 2016 en 7:29

La competencia que existe hoy es muy fuerte , JSF es una de muchas tecnologías y soluciones como Angular y React están haciendo que se use menos. Simplemente echa un vistazo a google trends y veras que la tendencia de JSF esta en declive. Eso no quiere decir que no se pueda usar . Simplemente que ahora mismo hay mas opciones y dependiendo de la situacion , opciones mejores.

Un saludo

Responder

Pedro dice
15 octubre, 2016 en 19:21

Llevo poco con JSF y estoy probando la versión 2.2, donde según leo, se puede configurar en el descriptor de despliegue de los JSF, si queremos que se ejecute en el cliente o en el servidor. Desconozco si esto es una característica nueva, pero entonces no debería verse a JSF como inferior frente a otros framework que se citan aquí que tienen esa posibilidad de ejecutarse en el cliente.

Responder

Cecilio Álvarez Caules dice
15 octubre, 2016 en 20:07

No es exactamente eso , lo que pueden hacer es salvar el estado en el cliente que es diferente , eso permite reducir memoria en el servidor pero aumenta el consumo de trafico http

Responder

aleon dice
24 septiembre, 2016 en 7:25

Saludos expertos, que me aconsejan como arquitectura de backend para capturar unos datos de un videojuego en unity y luego mostrar una metricas de uso y frecuencia del juego mas otros datos que google analitics no me entrega?
mil gracias.

Responder

Juan dice
19 agosto, 2016 en 18:32

Muy buen artículo. Yo he pasado recientemente de trabajar con frameworks javascript modernos y la verdad, no hay color.

Es terrible cuán acoplado está el código backend con el frontend, la lentitud que supone tener que compilar la mayoría de las veces para ver cambios en front, es tedioso.

Por no mencionar que si quieres renovar alguna parte de tu arquitectura ¡Ya no puedes! estás atado a ella de por vida, y mas si es un proyecto grande.

Un backend estilo API rest da mucha mas flexibilidad y tiene mucho mas sentido hoy en día, con clientes cada vez mas potentes, tanto web como nativos.

Responder

Cecilio Álvarez Caules dice
24 agosto, 2016 en 9:44

La plataforma de JavaScript tiene un futuro brillante ☺

Responder

Polymer dice
20 septiembre, 2016 en 15:50

Todo lo que dices es falso, se nota que JSF no lo has tocado ni en media semana, para ver cambios en el front no necesitas compilar... , utilizando JSF siempre vas a estar tocando JavaScript o CSS si quieres hacer una web en condiciones y

tocando lo último no estás atado… JSF renderiza por componentes puedes utilizar reglas de estilo sobre esos componentes o crear los propios, o directamente hacerlos en HTML 5 con passthrought, JSF es el futuro, JavaScript es bueno en el cliente pero no para la lógica de negocio, es un lenguaje horrible y a años luz para mantener y depurar cómo en Java, por no decir que Java es más rápido en el servidor que JavaScript y para la parte cliente y no tener que recargar la página al completo ya tienes ajax integrado.

Y eso de utilizar REST para la web me parece prehistórico, una cosa es para utilizarlo en plataformas nativas como Android o iOS, pero cuando existen plataformas que ya te lo dan todo mascado es una forma muy absurda de reiventar la rueda y perder enormes horas de tiempo y dinero, ah y no se dice "frameworks modernos" se dice código inmaduro que es muy diferente, el software no va por modas, no al menos si quieres hacer software serio.

Responder

Cecilio Álvarez Caules dice
20 septiembre, 2016 en 17:08

Bueno todas las opiniones son respetables

Responder

Polymer dice
24 septiembre, 2016 en 17:48

Las opiniones pueden ser respetables pero si son falsas alguien debería decirles algo, está muy de moda utilizar javascript en el servidor, pero cualquier ingeniero de software sabe que Javascript no es precisamente un lenguaje bien diseñado hace agua por todas partes y sin software mantenible y fácil de testear o con posibilidad de realizar TDD, estamos retrocediendo 20 años.

El único lenguaje que puede ser una alternativa es Python, la misma Google suele utilizarlo en situaciones donde no utiliza Java, pero estámos en las mismas es un lenguaje dinámico por tanto complicado hacer TDD y depurarlo, otro puede ser C# pero está muy enfocado en Window, pero si es un lenguaje a la altura de Java y por cierto PHP,JavaScript o Python en el servidor son muy muy lentos en comparación a la velocidad que alcanza Java con un programador que sabe patrones de diseño y entiende de manejadores de memoria como pools y como funciona el recolector de basura.

También recuerdo que en cualquier proyecto software el coste del 80% será su mantenimiento de hay que a nivel empresarial a nadie se le

ocurra utilizar lenguajes o tecnologías inmaduras.

Responder

Cecilio Álvarez Caules dice
25 septiembre, 2016 en 9:27

Siento discrepar , creo que JavaScript a nivel de cliente tiene un presente y futuro brillante.

Polymer dice
25 septiembre, 2016 en 19:54

Yo sólo me refiero al lado del servidor ya que muchos dicen que Java es lento y que JavaScript puede equipararse a Java en el lado del servidor siendo completamente falso, en el lado del servidor PHP, JavaScript y Python seguirán siendo lenguajes de segunda clase ya que son dinámicos y sin la madurez de Java a nivel de ingeniería del software.

Y si JavaScript no es que tenga un futuro brillante pero su presencia en el lado del cliente es esencial por ahora y al menos el diseño del lenguaje es apto para el lado del cliente hasta cierto punto.

El futuro que tiene por delante es el tiempo que tarden en sacar WebAssembly, recuerdo que detrás de el unen fuerzas Google, Mozilla y Microsoft, cuando de forma nativa para la web lenguajes más legibles tengan la misma velocidad JavaScript quedará en el olvido.

Un ejemplo https://webassembly.github.io/demo/ , y el rendimiento es sobre unas 20x frente a JavaScript, hay queda eso, eso si ya adelanto que JavaScript es lento de narizes en el cliente, cualquiera que haya hecho uso de el para crear un panel de administración con estadísticas del sitio lo sabe de primera mano la única ventaja es que por ahora es el único lenguaje de primera clase en la web, pero tiene los días contados :).

Responder

Cecilio Álvarez Caules dice
25 septiembre, 2016 en 20:27

Yo creo que en el servidor Java sigue ganando ☺

Will dice
9 junio, 2016 en 1:28

Creo que tomar gráficas que se basan en las tendencias de búsqueda es una forma muy superflua de abordar este tema y que puede dar resultados engañosos. si se combina ambas gráficas se ve como jsf sigue superando a spring mvc y no lo digo porque prefiera uno u otro, simplemente hay que evitar dar malas interpretaciones a los datos sino tenemos opiniones sesgadas por nuestras preferencias personales.

Responder

Cecilio Álvarez Caules dice
9 junio, 2016 en 7:21

Siempre es relativo y hay que ver como avanza todo primefaces ahora tiene unos proyectos muy interesantes con angular.

Responder

Cecilio Álvarez Caules dice
9 junio, 2016 en 7:23

de todas maneras la gráfica que a mi me ha salido es esta que sigue poniendo a Spring MVC por delante , pasame por favor un link a tu comparativa porque igual en otro sitio sale distinto , muchas gracias

Responder

Salim Castellanos dice
28 octubre, 2017 en 0:40

Hola, pues la verdad acabo de ver el link, y actualmente jsf esta empatado en búsquedas con Spring, y angular y react están demasiado distantes, es más, si te vas al mapa JSF gana en todos los países menos en uno. Saludos.

Responder

Cecilio Álvarez Caules dice
28 octubre, 2017 en 8:51

Buenas

Yo acabo de hacer una búsqueda por tendencias ☺ y ahora angular tiene más tirón.

Responder

franco chamas dice
6 mayo, 2016 en 21:59

son dos modos de trabajar totalmente distintos, si queres hacer un aplicativo lo mas rápido que se pueda jsf contribuye en gran medida, pero existen muchos

programadores que no les gusta ser configuradores, sino que hace uso de múltiples herramientas y lenguajes para hacer una funcionalidad innovadora, para hacer un simple ABM cualquier tecnología es buena, pero los desarrolladores java siempre buscamos nuevos desafíos.
de lo contrario estariamos hablando de webform + vb.net.

Responder

Elmer Morales dice
1 abril, 2016 en 2:42

Hola Cecilio, entonces que me recomendarías para realizar una página tipo amazon o una red social, es decir que tienen mucha concurrencia; saludos, felicitaciones por tu página.

Responder

Cecilio Álvarez Caules dice
1 abril, 2016 en 18:23

Spring MVC + Algún frawework MVC que te encaje ☺

Responder

Meyquel dice
23 marzo, 2016 en 20:48

Muy interesante el blog en general y energizante los articulos principalmente para aquellos que no dominamos el ingles perfectamente.
Mi apreciación sobre el tema es que todo depende de lo que las personas dominen. La mayoria de las personas que usan estos framework js provienen de php y el desarrollo web en general y no conocen mucho el mundo JSF. Cada cual habla de lo que sabe. Pero no creo que los desarrolladores JSF se estén pasando a framework js.
saludos

Responder

Cecilio Álvarez Caules dice
23 marzo, 2016 en 22:34

Ojala hubiera soluciones absolutas, creo que los frameworks JS son unos competidores muy fuertes , pero JSF también tiene su hueco.

Responder

william dice
21 enero, 2016 en 19:14

Hola,
Si se puede desarrollar aplicaciones grandes con bootstrap, jquery, angular ….. y spring mvc… depende de la arquitectura backend y la arquitectura frontend, pero sin dudas se pueden realizar componentes javascript (jquery, etc) las cuales se ejecutaran muy rapido en el cliente… Pienso en arq. SOA + microservices, y consumir todo en jquery, angular o lo que sea… con un buen layout… la seguridad se puede validar en ambos lados… Que curioso que en esta transnacional donde trabajo aun no usan JSF…

Responder

Cecilio Álvarez Caules dice
21 enero, 2016 en 19:37

A día de hoy va a menos claramente ☺

Responder

Luis dice
15 mayo, 2016 en 5:45

Buenas noches todos los framework basados en JS son utilizados para el frontend.Hoy en día ya la mayoria están llegando al servidor pero en la actualidad el performance no es comparable con JSF.
JSF es un framework para aplicaciones web transaccionales.Este se vale de PrimeFaces para decorar mejor su presentación. Ahora salió otro candidatos para su decoración FacesBostrup.
Solo son tecnologías para el frontend. Ahora el manejo de la paginación de una grilla siempre va ser mas lento en formularios basados en js, del lado del cliente. La mejor tecnología para el manejo de paginación la tiene jsf PaginationHelper. Una paginación hecha en primeces , NodeJS, Angular es muy lenta porqué. Primero se leen toda la tabla y depues hacen la paginación mientras la que utiliza jsf la rewaliza por el numero de registros que qquieres mostras. sin importar si la tabla tiene 10 millnes de registros, esta te muestra la grilla en menos de un 3 segundos.
Yo soy un ongeniero experto en tecnologia java EE y trabajo con JSF 2.x, JPA 2.1, EJB 3.x, Primefaces 4,5.x, EclipseLink, GlassFish, JBoss, etc. Lo importante de todo el tema que se trata aquí es que el performance se

consigue con la configuración, buena programación las escogencias de las componentes.

Responder

Cecilio Álvarez Caules dice
15 mayo, 2016 en 12:35

No hay una solución unica también puedes acabar paginando con Angular o con REact o con lo que necesites. Para mi primefaces es de lo más avanzado que hay en el mundo JSF y esta integrando soluciones tipo Angular con el.

Responder

Willson dice
12 julio, 2018 en 20:21

Yo trabajo con JSF y es lo mejor que he visto vengo desde los años 80 desarrollando desde D.O.S. y JSF es lo mejor que he visto no tiene limitaciones cuando se sabe trabajar.

Responder

Cecilio Álvarez Caules dice
20 julio, 2018 en 11:23

JSF tiene sus ventajas al ser un standard

Unknown dice
7 enero, 2016 en 19:32

Tiene mucho futuro. En una encuesta en DZone (https://dzone.com/articles/poll-what-java-jvm-frameworks-do-you-use) en Agosto pasado, JSF lideró con 34.5% (463 votos) frente al 34.2% (459 votos) de Spring MVC. Si vemos encuestas anteriores, Spring MVC lideraba al menos por 5-7% de diferencia. Si bien la muestra

es pequeña, es un pequeño reflejo del trabajo que se viene haciendo en JSF. JSF 2.2 y su compatibilidad con HTML5 han tocado fondo en los programadores (me incluyo).

He trabajado con JSF y Spring MVC y ambos me parecen soluciones válidas para la mayoría de casos (salvo real-intensive real time apps, en donde no dudo en usar NodeJS). Decir a estas alturas que JSF es más pesado está de más. Zeef, la página en donde puedes buscar recursos de todo tipo, está hecha en JSF. ¿Es lenta? Para nada. Parece que el fanatismo de algunos programadores por X framework hace que menosprecie a la competencia.

Lo que no me gusta del mundo de JSF es PrimeFaces. Los temas CSS por defecto son horribles, y para editarlos tienes que saber CSS intermedio; si usas un framwork tipo Bootstrap, la integración no se llevará a cabo y se verá feo estéticamente. Por ello siempre uso Bootstrap para mis aplicaciones, incluso sus controles JS como acordeón, modales, carruseles, tabs, etc.

Para resumir:

Aplicaciones pequeñas/Sitios web: Play, Spark
Aplicaciones pequeñas, medianas y grandes: JSF, Spring MVC, Struts

Responder

Cecilio Álvarez Caules dice
7 enero, 2016 en 23:24

Yo tengo más dudas cada día JavaScript es más potente en el lado cliente y va a ser un competidor muy duro. No digo que JSF este acabado pero si que va a tener más problemas.

Responder

Unknown dice
10 enero, 2016 en 20:57

Yo tampoco tengo duda de eso. Y no se está haciendo más fuerte solo en el lado cliente, NodeJS está creciendo a un ritmo imparable. Yo creo firmemente que el futuro a mediano plazo será JavaScript, incluso puede dar batalla a Java/NET en el mundo Empresarial.

Como dijiste en un artículo, Java avanza lento y aunque desde una perspectiva es bueno para la empresa, creo que tiene su contraparte y es que si lo comparas con otros lenguajes como C# o Python, a mi gusto, Java como lenguaje, queda rezagado.

Es aconjonante ver como Oracle no invierte lo suficiente en Java para prestarle más atención a sus productos cloud. La JCP y su benditas JSR son los que lo mantienen en auge, porque Oracle poco (por no decir nada) aporta. Un futuro muy diferente hubiese sido si IBM ganaba la batalla por la compra de SUN ☺

¡Saludos!

Responder

Cecilio Álvarez Caules dice
10 enero, 2016 en 21:42

Muchas gracias por el aporte ☺ comparto tu opinión de que node esta creciendo muy rápido y va a competir de tu a tu en el futuro

Responder

Joan Costa dice
1 noviembre, 2015 en 14:03

¿que tecnología de presentación me aconsejaríais para implementar aplicaciones empresariales de gestión (contabilidad, gestión logística, etc.) en la intranet de la empresa?

Responder

Cecilio Álvarez Caules dice
16 noviembre, 2015 en 14:07

Es una pregunta muy abierta, pero sobre todo elegiría una tecnología de largo recorrido en el tiempo.... JSF podría ser un ejemplo o también Spring MVC ... dependerá mucho de lo complejo de la interface de usuario . A mayor complejidad mas encaja JSF. Hoy por hoy los frameworks de JavaScript de cliente son también una opción pero hay que empezar poco a poco con ellos . Yo ahora mismo si tuviera que elegir selecionaria Spring MVC + bootStrap

Responder

Jairo dice
8 septiembre, 2015 en 15:27

La velocidad de implementar Spring mvc (Angular+bootstrap), que consuman recursos del cliente es muy grande y superior a Struts 2.
Java JSF este destinado a mejorar o desaparecer!!!

Responder

Mitsu Gami dice
4 septiembre, 2015 en 16:58

Entre el desarrollo moderno de aplicaciones web y el enfoque de desarrollo de Java EE (JSF) hay sin duda alguna un gran vacío en donde hace falta construir un puente para unir ambos.

Yo soy desarrollador con Java EE/JavaFX pero también me soy un diseñador/UX aficionado. Me gusta siempre que mis aplicaciones tengan una vista que sea adapte a los estándares modernos y por ende una experiencia de usuario bastante agradable y creo que aquí es justamente en donde viene el conflicto con JSF, donde

te limita de cierta forma la comunicación entre frontend y backend, en concreto, no te permite personalizar el comportamiento.

JSF es un framework que te abstrae de las tareas frontend comunes para que solo te centres en el backend. Por ejemplo, en lugar de hacer un script JS que haga una llamada AJAX JSF lo hace por tí tan solo mediante:

Personalmente nunca me ha gustado por más simplistas que sean, este tipo de frameworks porque no suelen adaptarse al cambio de los estándares frontend, como por ejemplo las SPA. Spring MVC por ejemplo, es mucho más amigable en este aspecto que JSF. Creo que fue un 'error' por así decirlo de SUN crear este framework. Si bien abstrae de tareas comunes y nos ahorra tiempo de desarrollo, te limita como programador/diseñador.

Por otro lado, la plataforma Java EE es realmente espectacular. JPA para el manejo de base de datos, JTA para transacciones, EJB como beans transaccionales y sus métodos asíncronos, CDI y su inyección de dependencias, calificadores, disposers, producers, eventos, JMS como servicio de mensajería, JAAS para el manejo de seguridad, y más. Todas estas son tecnologías muy maduras, testeadas y utilizadas durante muchos años que nos proveen seguridad y eficiencia en nuestras aplicaciones.

Por consiguiente, creo que hay actualmente 2 opciones:

– Utilizar Java EE (EJB, JPA, JMS, CDI, etc.) únicamente para backend y Angular, BackBone, CofeeScript, etc. para el cliente teniendo como vínculo entre ambos JAX-RS (Restful).

– Utilizar Java EE y Spring MVC como framework MVC y las tecnologías cliente que sean necesarias.

Personalmente elijo la segunda. Tengo mucha libertad en el frontent y puedo hacer lo que desee. Pero el uso de RESTs tiene un impacto no tan importante en el cliente que hay que tener en cuenta.

De todos modos, tengo 21 años así que carezco de la experiencia de muchos aquí y probablemente mi punto de vista esté equivocado.

Saludos.

Responder

Cecilio Álvarez Caules dice
5 septiembre, 2015 en 7:34

Es un buen análisis , existen siempre más opciones como optar por JSF y PrimeFaces en la presentación , pero básicamente las dos que tu expones son

las principales y la decisión esta muy ligada a la cultura empresarial de la organización

Responder

Emiliano dice
13 agosto, 2015 en 19:16

Jsf me gustó mucho, cuando era estudiante. Cuando llegue al ámbito laboral, trabajamos con Ajax y Jquery y al ver que este último no se podía integrar con JSF, le dije adiós. No me agrada que el código del cliente este muy amarrado a un framework, suficiente con Jquery.

Responder

Manuel Gonzalez dice
11 junio, 2015 en 9:11

hola a tod@s,
No se puede aplicar la misma solución a diferentes problemas. Las necesidades de una aplicación orientada a internet en la mayoria de los casos dista de una aplicación empresarial. Personalmente conozco jsf desde su versión 0.9 allá por el 2004-2005, pero con el tiempo para hacer aplicaciones empresariales me decanté por zkoss por considerarlo mas ligero y 100% ajax, además permitiéndome programar toda la lógica de comportamiento en java.

Hoy para la parte publica (internet) utilizamos angular js simplemente porque mediante json (apache cxf rest, jpa (hibernate), spring security y spring mvc en servidor), me permite dar mucho mejor rendimiento con menos recursos y mas escalabilidad en diseño.

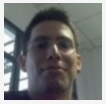
El backend lo mantenemos con zkoss, BPM de activiti , servicios soa, brms ....

Personalmente recomiendo que en aplicaciones de gestión (intranets/extranets) se utilicen los frameworks como jsf, gwt,zkoss agilizan el desarrollo de potentes aplicaciones empresariales y permiten trasladar esa ansiada funcionalidad que tenían antaño las aplicaciones RIA de escritorio.

Pero si lo que vamos a construir es un portal con un elevado acceso de usuarios y que además el mismo diseño sea multi-canal (movil,web, tablet...) y donde los formularios no son demasiado complejos, recomendaría la utilización de angular js o jquery....

Por ello no creo que esté decayendo JSF y otros frameworks del lado de servidor, simplemente se está racionalizando su uso con la aparición de mejores frameworks en javascript.

Responder



Sebastián Garcia dice
10 junio, 2015 en 23:57

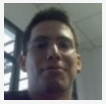
Interesante entrevista a uno de los creadores de zeef.com donde se debate este tema http://www.adam-bien.com/roller/abien/entry/a_java_ee_startup_filtering

Responder

Cecilio Álvarez Caules dice
11 junio, 2015 en 9:11

Gracias por el aporte ☺

Responder



Sebastián Garcia dice
10 junio, 2015 en 23:56

Hola, primeroque nada muy interesante tu blog. Con respecto a este artículo no estoy de acuerdo con lo que mencionas. Si bien es cierto que los frameworks JS hoy por hoy están ganando popularidad, no quiere decir que JSF este perdiendo popularidad. Muchos criticaron JSF en su momento (versiones 1 y 1.2) pero desde la versión 2.0 y especialmente 2.2 JSF mejoró muchisimo en un montón de aspectos. Siempre se critico que es un framework Server-centric (del lado del servidor) y que consumia mucha memoria al mantener el estado de la vista en el servidor, pero gracias al PARTIAL-STATE-SAVING hoy por hoy JSF tiene una performance igual o superior a muchos frameworks JS, sin contar con todas las ventajas que tiene: Componentes, Converters, Validators, etc. Y gracias a librerías como primefaces, primefaces-extensions y omnifaces JSF se potencia mucho más (Client side validations con soporte de bean validations y un largo etc.).

Ejemplos de uso de JSF en web apps publicas es zeef.com no solo usa JSF sino varias tecnologías de Java EE.

Para no extenderme más les dejo y recomiendo leer las 3 partes de este artículo en inglés donde se evalúa la perfiomrance de varios frameworks para Java

http://content.jsfcentral.com/c/journal/view_article_content?cmd=view&groupId=35702&articleId=73398&version=1.8#.VXiyKkYaN9w

Responder

Cecilio Álvarez Caules dice
11 junio, 2015 en 9:12

gracias por los aportes ☺

Responder

Daniel Villegas dice
9 junio, 2015 en 15:52

Gracias por la nota, esta excelente, muy bueno el debate que se armo, desde mi punto de vista, el futuro esta en separar totalmente el Fontend del Backend, ya sea

con Angular, Backbone, Polymer o cualquier otro Framework MVC Js del lado del Cliente, y del lado del Servidor Java, Spring MVC, NodeJs, etc.

De esta forma se puede separar totalmente el desarrollo de ambas partes, y enfocarse de lleno en el negocio y en la la vista, cada uno con su lógica.

De todos modos con un amigo estamos desarrollando en pequeño Faces "framework", en realidad es una extensión de JSF, por ahora, creamos un par de componentes, como, dataTable, commandButton, popup, paneles, etc, son todos component composition, lo subimos a git, sin mucha explicación ni nada, la url es:

https://github.com/Clevcore/clevcore-faces

Lo estamos usando en un proyecto privado, al estilo de lo que nombraron anteriormente, es decir, sistemas de gestión del rubro de la medicina, que tranquilamente puede estar dentro de la Intranet, como así de Internet, aunque normalmente estos proyectos están dentro de una Intranet.

También les cuento que hace dos años, realizamos una red social, utilizando JSF, creando componentes compuestos, como firma, comentario, etc, y resulto un dolor de cabeza realizarlo, en ese momento me pregunte, debería existir una mejor forma, y actualmente creo que seria usar otras tecnologías para manejar el fontend y backend, pero por mi corta experiencia creo que el futuro esta en separar totalmente cada uno, y comunicarlos, no se con .json

Responder

Stanley dice
8 junio, 2015 en 2:28

Particularmente me gusta por la productividad con la que se trabaja con JSF2 y más usando Primefaces. Algo que en mi caso particular considero una fortaleza para JSF es que el manejo de validaciones se hacen en el servidor lo que le brinda mayor seguridad a la aplicación.

Responder

Cecilio Álvarez Caules dice
8 junio, 2015 en 11:51

Es cierto que tiene un sistema sólido de validaciones de servidor ☺

Responder

Martin dice
6 junio, 2015 en 2:38

Es cierto jsf1 o jf2 version reciente combinado con primafaces esta bueno, pero el cliente pierde todo el control en el frontend debido que esos componentes los maneja el server, tambien apuesto por spring mvc, tengo bastante experiencia en struts2 me encanta es rapido y sencillo, pero quiero probar spring mvc a ver que tal resulta.Saludos desde Argentina.

Responder

Cecilio Álvarez Caules dice
7 junio, 2015 en 11:33

spring mvc tiene un buen futuro

Responder

Carlos Apablaza dice

25 mayo, 2015 en 19:26

Hola Cecilio,

Primero que todo, muchas gracias por compartir tus conocimientos con todos. Ahora respecto al futuro de JSF yo creo que compararlo con tecnologías en cliente es bastante superficial(no critico tu visión, sino que invito a expandirla). Según mi experiencia(tanto jsf como angular, spring mvc, otros..) cada tecnología tiene su ventaja y debemos tomar muy en cuenta las competencias del mercado. Por ejemplo, la simplicidad que nos entrega JSF(Primefaces, MyFaces,etc…) es muy alta y esto claramente permite mantener alto enfoque en la solución y no en la tecnología usada. Ahora bien, frameworks como angular, backbone… nos proporcionan toda una solución MVC en cliente, pero "cuantos buenos desarrolladores existen en el mundo JS". No es una critica, pero considero que aumenta el grado de dificultad para los desarrolladores y en consecuencia la curva de aprendizaje es mucho mas alta.
Ahora no podemos dejar de lado, otras tecnologías emergentes como lo son NodeJS(JS en servidor), Bases de datos NoSQL, etc… Estas también proporcionan soluciones(ExpressJS) que afectan directamente a la popularidad de JSF.

Yo creo que todo depende del enfoque requerido:

– Si necesitas soluciones intranet para empresas, usar JSF es una gran alternativa por su efectividad, simpleza. Pero su rendimiento no es comparable con tecnologías de vanguardia(NodeJS)
-Si necesitas soluciones para internet, debes dejar de pensar en una sola tecnología dado que en la industria no se maneja una sola. Existen diversas opciones para cada requerimiento, es decir, puedes construir algo con JEE, Rails, Python, AngularJS, NodeJS,etc…

Finalmente, todo depende de lo requerido por tu cliente, producto o servicio.

Espero sea de utilidad lo comentado… saludos y reitero mis felicitaciones por tu blog.

Responder

Cecilio Álvarez Caules dice
26 mayo, 2015 en 20:55

Yo creo que todo tiene su hueco, JSF ha sido utilizado en intranets con bastante éxito. Pero hoy por hoy pienso que el universo JavaScript en el lado cliente esta sumando muchos apoyos por parte de la comunidad y eso le concede una posición digamos de “futuro” muy sólida

Responder

Aritz dice
27 mayo, 2015 en 18:28

Buenas tardes,

No entiendo por qué insistís en intranet/internet. JSF es un framework web orientado a componentes y el cual mantiene un estado de lo que sucede en el cliente en el lado servidor. Esto tiene sus ventajas y sus inconvenientes. La ventaja es que el servidor es capaz de conocer en todo momento cual es el estado de los componentes. La integración ajax y la conversión de objetos de cliente a objetos servidor es poco menos que automática y la validación de los campos formulario también está integrada para el lado servidor. Por lo tanto las ventajas de JSF son simplicidad + seguridad.

El gran inconveniente bajo mi punto de vista es la necesidad de mantener este estado actualizado: La filosofía JSF en una vista (url) es básicamente notificar al servidor cuando se desea que se produzca un cambio y que el servidor vuelva a renderizar el DOM, todo esto por Ajax. En un framework orientado a requests, se modifica el DOM de la página en el cliente y cuando deseemos procedemos a lanzar el request. Por lo tanto obtenemos una opción más dinámica en el segundo caso, pero no creo que la

diferencia resida en intranet/internet, si no en el tipo de aplicación. No programaría una aplicación cuyo peso resida en el lado cliente en JSF (con opción de trabajar offline o con poca conectividad, por poner un ejemplo). Pero JSF es perfectamente válido para aplicaciones de gestión y demás, que tengan una lógica importante en el lado servidor y sean relativamente simples en cliente.

Un saludo

Responder

Cecilio Álvarez Caules dice
27 mayo, 2015 en 21:24

Es cierto que la seguridad es uno de sus puntos fuertes . Ojalá hubiera una respuesta completamente válida para todo pero depende mucho de las situaciones

Responder

Gladys Medina Jiménez dice
28 mayo, 2015 en 18:24

¡Buenas tardes!

Estoy totalmente en sintonía con lo que dice Aritz. En mi opinión, pensar que JSF es válido solo para aplicaciones de Intranet es subestimar la potencia de este framework.
Dentro del ámbito de aplicaciones de gestión, donde hay una lógica de negocio considerable (por ej.: entrada/salida de mercancía, gestión de stocks, venta de productos), yo les aseguro que JSF es una maravilla y más aún si lo combinamos con otros que facilitan la vida del equipo de desarrollo: Spring (para la gestión de sesiones a la base de datos, para el control de transacciones, etc..) e Hibernate para la persistencia en base de datos relacionales. No olvidemos que éstas son más fiables desde el punto de vista de la consistencia de datos que las denominadas NoSQL, lo que las hace ideales para este tipo de aplicaciones.

Saludos,

Cecilio Álvarez Caules dice
29 mayo, 2015 en 7:09

No lo dudo pero al ser un framework basado en componentes va a tener una dura competencia con los nuevos standares de HTML5 que se denominan web components y que entran en competencia directa con su filosofia. Lo mismo le va a pasar a ASP.NET Web Controls

Oscar Calderon dice
11 junio, 2015 en 20:50

A lo que me imagino que se refiere Cecilio (corrijame si me equivoco) es a la demanda de usuarios que tendría. Hablando de aplicaciones intranet en la mayoría de los casos entendería que hablamos de aplicaciones para empresas, que serían utilizadas por el personal de una empresa únicamente. Pudieran ser 2 usuarios, 5, 20, 100 por dar un

ejemplo, teniendo límite en el tamaño de personas que conforman la empresa por ejemplo, lo cual es algo fácilmente manejable.

Pero hablar de una aplicación en internet implica que podría ser utilizada por 10000, 100000, millones de personas, es decir una demanda muchisimo mayor en la cual se necesitan aplicaciones que sean capaces de aprovechar al máximo los recursos y poder atender cientos de miles de peticiones de forma concurrente, algo para lo cual se han especializado elementos como nodejs por ejemplo.

Responder

Cecilio Álvarez Caules dice
15 junio, 2015 en 9:08

Sobre todo me refiero a que una aplicación en internet puede tener un acceso remoto lejano en donde el trafico de red esta muy reducido y hacer peticiones al servidor pueda penalizar. En estos casos estar recargando continuamente la página penaliza mucho.

nelson dice
15 mayo, 2015 en 17:12

Me gustaría empezar a aprender un poco sobre SpringMVC pero, hay tantas cosas que no se por donde empezar. De hecho, no se qué es lo que debo buscar. Me gustaría saber si me podrías recomendar un path de cosas puntuales.

Responder

Jesus Perales dice
15 mayo, 2015 en 16:54

Muy bueno el blog pero hace falta los botones para compartir sus articulos , los botones de la izquierda algunos no funcionan.

Responder

Cecilio Álvarez Caules dice
16 mayo, 2015 en 7:36

Gracias por la información he actualizado wordpress a la última versión yo creo que ahora funciona ☺

Responder

Gladys Medina Jiménez dice
13 mayo, 2015 en 14:42

Pues ya veremos qué pasa. Mientras tanto estoy encantada con JSF 2, Spring, Hibernate y, cómo no, con primefaces…

Responder

Cecilio Álvarez Caules dice
13 mayo, 2015 en 16:13

Si hay que esperar un poquito a ver como avanza todo . Parece que JavaScript se comerá el lado cliente .... pero hoy por hoy todavia no lo ha conseguido

Responder

SANTIAGO ROJAS MANIOS dice
4 mayo, 2015 en 1:12

Me encanta su blog, arquitecto.
Y quisiera tener un poco mas de tiempo para poder ponerme al día y leer todas las entradas. ☺
Ahora me gustaria, me diga que herramienta usas para hacer los diagramas que colocas, es
que se ven sencillos y muy puntuales.

Responder

Cecilio Álvarez Caules dice
4 mayo, 2015 en 8:31

Onmigrafle para mac ☺

Responder

Harry Izquierdo dice
27 abril, 2015 en 4:23

no que con PrimeFaces Extensions se puede tener los controles del lado del cliente?

Responder

Harry Izquierdo dice
27 abril, 2015 en 4:22

PrimeFaces Extensions

Responder

Cecilio Álvarez Caules dice
27 abril, 2015 en 21:18

gracias por el aporte no los había ni mirado … cual ha sido tu experiencia?

Responder

Ivan Emilio Garcia dice
24 abril, 2015 en 17:27

Yo he visto que primefaces ha agregado nuevos componentes y ha creado unos templates (de paga) y temas nuevos que se ajustan mucho el diseño actual de aplicaciones. Sin embargo, tiene mucho empuje los frameworks javascript, esperemos que JSF siga mejorando y las versiones de myFaces y mojarra sean mas rápidas.

Responder

Cecilio Álvarez Caules dice

26 abril, 2015 en 13:37

Vamos a ver como avanza todo y luego se podrán tomar más decisiones ☺

Responder

junagel dice
23 abril, 2015 en 13:10

Pues todo apunta a una perdida de popularidad, puede ser por la aparición de otros frameworks que utilizan mas la parte de cliente como dices. Es un problema para los que estudiamos esta tecnología, y ahora me pregunto, que pasa con los frameworks para jsf? proyectos tan fuertes como primefaces, icefaces, etc. que te facilitan tanto la vida.

Responder

Cecilio Álvarez Caules dice
24 abril, 2015 en 8:24

Yo croe que seguirán teniendo su hueco pero habrá que ver si soportan el empuje de Javascript. Por ahora van a tener que compartir su cuota de mercado con más competencia.

Responder

Alarcon dice
4 mayo, 2016 en 15:46

La respuesta a lo que pasa con frameworks como primefaces es que ellos ya estan migrando a esas tecnologias: primefacesUI y primefacesNG.

Responder

Cecilio Álvarez Caules dice
5 mayo, 2016 en 19:47

Gracias por el aporte ☺

Responder

Trackbacks

El ecosistema Java y su adaptabilidad - Arquitectura Java dice:
23 septiembre, 2016 a las 13:50
[...] artículos relacionados : ¿Tiene futuro JSF? , Spring vs Java [...]

Responder

¿Es el fin de los servidores Java EE? - Arquitectura Java dice:
1 abril, 2016 a las 13:22
[...] artículos relacionados: ¿Tiene futuro JSF? , Java EE vs Spring [...]

Responder

Deja un comentario

Tu dirección de correo electrónico no será publicada.

Comentario

Nombre

Correo electrónico

Web

PUBLICAR COMENTARIO

Este sitio usa Akismet para reducir el spam. Aprende cómo se procesan los datos de tus comentarios.

BUSCAR

Buscar en esta web

Cursos Gratuitos

Introduccion
Spring Boot

Introducción
TypeScript

Introduccion
JPA

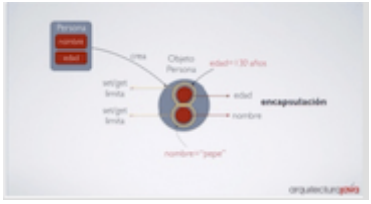
Java Herencia

Java JDBC

Servlets

Mis Cursos de Java

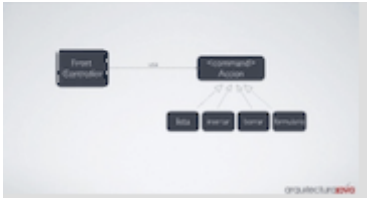
Programación Orientada a Objeto en Java



Java APIS Core

Java Web

Pack Java Core



Arquitectura Java Solida con Spring

POPULAR

Spring REST CORS y su configuración

Spring Boot JSP y su configuración

Arquitecturas RESTFul y agregados

Curso Spring Boot y sus tecnologías

@RepositoryRestResource y Spring Framework

Single Page Application y REST

Spring JdbcTemplate y el principio DRY

El Principio de Substitución de Liskov

Eclipse JPA y clases de dominio

El modelo Entidad Relacion con DBDesigner

CONTACTO

contacto@arquitecturajava.com

LO MAS LEIDO

Java Currying y la programación funcional

¿Qué es Spring Boot?

Curso Spring Boot y sus tecnologías

Java Constructores this() y super()

Usando Java Session en aplicaciones web

Java Iterator vs ForEach

Introducción a Servicios REST

Ejemplo de Java Singleton (Patrones y ClassLoaders)

REST DTO y JSON Arquitecturas Web y objetos

Comparando java == vs equals

Usando el patron factory

Java Override y encapsulación

Ejemplo de JPA , Introducción (I)

Uso de Java Generics (I)

REST JSON y Java

¿Cuales son las certificaciones Java?

¿Qué es Gradle?

¿Qué es un Microservicio?

El patrón de inyección de dependencia y su utilidad

Angular ngFor la directiva y sus opciones

Mis Libros

Java Stream forEach y colecciones

¿ Que es REST ?

Spring MVC Configuración (I)

Spring Boot JPA y su configuración