Mejores prácticas de Camunda

Usando una pila Greenfield

Camunda es muy flexible y se puede conectar a la arquitectura de su elecció
brinda una serie de decisiones importantes que tomar. Pero si no tiene requ
especiales de arquitectura, le recomendamos que simplemente use esta pila
propuesta.

Uso de una pila Greenfield ¿

? best-prac

Introduciendo el Pila Greenfield

La pila greenfield es actualmente una recomendación para los **desarrolladores de Java** .

> Estamos trabajando en una recomendación nueva para usar diferentes pilas (como .NE
> ahora, consulte, por ejemplo, esta publicación de blog sobre cómo usar Camunda en ur
> [1] .

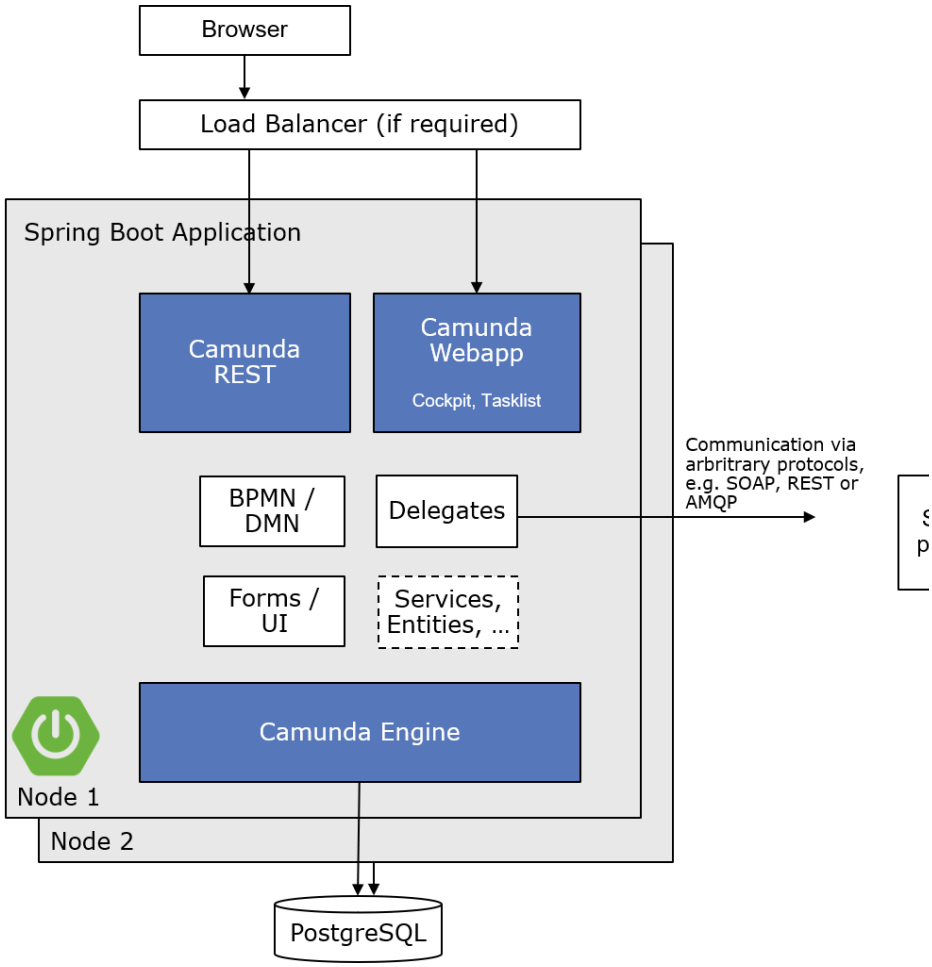
1. Inicie su proyecto de automatización de procesos como una aplicación Spring Boot [2] , utilizando Camunda Spring Boot para la API REST [3] y las aplicaciones web [4] .
2. Construye tu proyecto con Maven [5] .
3. Use Eclipse [6] como IDE,
4. Use la Edición Enterprise [7] de camunda-engine y camunda-webapp.
5. Use OpenJDK 11 [8] como tiempo de ejecución de Java.
6. Modele los procesos con Camunda Modeler [9]
7. Agregue sus modelos de proceso y el código de Java al proyecto
8. Utilice la base de datos java basada en archivos H2 para el desarrollo. Nos **oponemos enérgicam** desarrolladores comparten la misma base de datos durante el desarrollo ya que esto puede cond de problemas.

Ejecutando la aplicación de proceso en **producción** :

1. Utilice PostgreSQL [10] , o la base de datos que ya opera.
2. Asegure el Rest-API. Mira este ejemplo [11]
3. Ejecute la aplicación de proceso copiando el archivo jar en el servidor y comience con `java -jar YourProcessApplication.jar`

Comprender la arquitectura de la pila

La arquitectura básica con esta pila se muestra en el siguiente diagrama:



Ver más detalles sobre decisiones de arquitectura en 🔒 Mejores prácticas aún no publicadas .

La aplicación Spring Boot contiene un servidor Tomcat incorporado para servir las interfaces externas. proceso está incrustado. Se puede construir un clúster iniciando más instancias usando la misma base

Comprender nuestra motivación para la pila

Si bien pasamos por discusiones largas y detalladas para llegar a esta recomendación, **no** significa que superior a las pilas alternativas. Por lo tanto, aún puede sentirse bien si toma otra ruta (consulte Decic obtener opciones alternativas). Pero para las mejores prácticas, queríamos dar **exactamente una** rec para todos nuestros clientes que no tienen requisitos especiales en la pila.

Decidimos por esta pila porque:

- Todos los componentes son de código abierto y están fácilmente disponibles.
- Spring Boot es actualmente la forma más adoptada de aplicaciones Java buidlind
- PostgreSQL tiene un excelente historial de rendimiento.
- Las aplicaciones Spring Boot son fáciles de personalizar y de implementar en entornos de prueba y en las instalaciones o en la nube.

Hay **ventajas al usar la pila greenfield** :

- **Menos decisiones:** dependiendo de su experiencia con el cosmos de Java, las decisi una pila pueden no ser fáciles de tomar. Entonces, si no tiene requisitos especiales, y siga un camino conocido.
- **Menos esfuerzo de aceleración:** proporcionamos distribuciones preempaquetadas documentación y guías de instalación para la pila propuesta.
- **Probado:** Muchos de nuestros clientes usan esta pila con gran éxito.
- **Más documentación y mejores prácticas:** para que no tenga que escribir su propi extensa, solo señale los documentos de Camunda.
- **Soporte más fácil:** pedir ayuda se vuelve mucho más fácil ya que no tiene que expli configuración en detalle.
- **Plantillas y ejemplos de** proyectos **:** proporcionamos plantillas de proyectos (como Maven), que pueden adaptarse directamente a usted.

Personalizando la pila

Está perfectamente bien elegir otra pila si tiene buenas razones para ello. Por ejemplo, es posible que

- Use la **base de datos** que ya opera. Agregue el controlador de base de datos a pom.xml y configur application.yml según corresponda.
- La caída de la `camunda-bpm-spring-boot-starter-rest` del `pom.xml` si no lo necesita.
- Suelte el `camunda-bpm-spring-boot-starter-webapp` de `pom.xml` si inicia un servidor separado caso, establezca la `deployment-aware` configuración en `true` .

Comenzando con la pila Greenfield

1. Verifique los requisitos previos:
   - Instalar OpenJDK 11 [8] .
   - Instalar Camunda Modeler [9] .
   - Instalar Eclipse [6] . Recomendamos el último "Eclipse IDE para desarrolladores Java".
     - Active la actualización de sincronización de archivos del espacio de trabajo utilizando enlace [12] para mejorar la interacción de Eclipse y Camunda Modeler.
     - Agregue Camunda Assert a sus favoritos de Eclipse Content Assist [13] .
   - Verifique su acceso de red a Camunda Nexus [14] para descargar artefactos Maven.
   - Como cliente empresarial, verifique que tenga a mano las credenciales de su empresa para inic versiones empresariales.
2. Crea tu proyecto
   - Usa el Arquetipo de Maven [15] 'camunda-archetype-spring-boot' para crear un nuevo proyecto
   - Modele un proceso con Camunda Modeler y guárdelo debajo `src/main/resources` .
   - Ejecute el proyecto desde su IDE: comience `CamundaApplication` como aplicación Java.
   - Juega con tu proceso usando la aplicación web Camunda (Tasklist and Cockpit).
   - Empaquete su aplicación con `mvn clean install` .
   - Traiga el archivo jar a su servidor de prueba o producción y comience allí.
   - Puede configurarlo o integrarlo en una tubería de entrega continua existente.
3. ¡Aprenda más sobre las definiciones de proceso de prueba y agregue pruebas unitarias a su proce

¡Mira el tutorial Camunda BPM Spring-Boot [16] para obtener más detalles sobre cómo comenzar !

Links

[1] https://blog.bernd-ruecker.com/use-camunda-without-touching-java-and-get-an-easy-to-use-rest
orchestration-and-workflow-7bdf25ac198e
[2] https://spring.io/projects/spring-boot
[3] https://docs.camunda.org/manual/7.11/user-guide/spring-boot-integration/rest-api/
[4] https://docs.camunda.org/manual/7.11/user-guide/spring-boot-integration/webapps/
[5] https://maven.apache.org/download.cgi
[6] https://eclipse.org/downloads/
[7] http://camunda.com/bpm/
[8] http://jdk.java.net/11/
[9] https://camunda.org/download/modeler/
[10] http://www.postgresql.org/
[11] https://github.com/camunda-consulting/code/tree/master/snippets/springboot-rest-api-basic-a
[12] http://stackoverflow.com/questions/4343735/avoiding-resource-is-out-of-sync-with-the-filesyste
[13] https://github.com/camunda/camunda-bpm-assert/blob/master/camunda-bpm-assert/READM
camunda-bpm-assert-to-eclipse
[14] https://app.camunda.com/nexus/
[15] https://docs.camunda.org/manual/7.11/user-guide/process-applications/maven-archetypes/
[16] https://docs.camunda.org/get-started/spring-boot/

Descargo de responsabilidad y derechos de autor

Printed November 19, 2019. Applies to Camunda **7.11**. Any feedback? **best-practices@camur**