Pensieri sul futuro di Jakarta EE senza "javax"

Questa settimana Mike **Milinkovich** ha annunciato che **Oracle** e la **Fondazione Eclipse** hanno concordato che il pacchetto **javax** non può essere evoluto dalla comunità **Jakarta EE** e che i marchi **Java** non possono essere utilizzati nelle specifiche **Jakarta EE**. Quanto è importante per la comunità Jakarta EE? Jakarta EE può sopravvivere senza lo spazio dei nomi javax?

Di sicuro, sarebbe molto meglio **evolvere le API esistenti** nei pacchetti **javax** senza alcuna restrizione. Ma abbiamo quello che abbiamo. C'erano anche ragioni per farlo in questo modo. D'altra parte, finalmente, le **incertezze** con i marchi **Java** sono sparite. È molto importante perché il progresso di **Jakarta EE non è più bloccato**. Le restrizioni del pacchetto Javax tracciano una linea chiara che **separa** Java EE e Jakarta EE, separando passato e futuro. Javax diventa un'eredità. Tutte le innovazioni apparterranno allo spazio dei nomi jakarta. Ora è chiaro che i pacchetti javax non possono essere modificati e la comunità ha bisogno di trovare un modo per evolvere le specifiche. E ci sono alcune opzioni su come può essere fatto. Dobbiamo **massimizzare** la **compatibilità** con Jakarta EE 8 per le versioni future senza soffocare l'innovazione. È quanto concordato dal Comitato per le specifiche dell'EE Jakarta.

A mio parere la cosa migliore sarebbe fare un **big-bang** javax->jakarta rinominare il pacchetto per sciogliere le mani della comunità e consentire modifiche ed evoluzione delle API. **Spezzerà** la **retrocompatibilità** che è sempre stata una parte importante della piattaforma Java EE e non dovremmo dimenticarcene. Esistono migliaia di applicazioni Java EE e non sarebbe saggio rimuovere i requisiti di compatibilità con le versioni precedenti. Una buona opzione è **creare uno speciale profilo di compatibilità** con le versioni precedenti nella piattaforma Jakarta EE. Questo profilo dovrebbe contenere un'API Java EE 8 bloccata e consentirà di eseguire applicazioni Java EE 8 su versioni future della piattaforma Jakarta EE. Questo profilo può essere facoltativo per consentire ai nuovi potenziali fornitori di Jakarta EE di concentrarsi solo sulle innovazioni, ma sono sicuro che tutti i grandi attori come Oracle e IBM lo supporteranno comunque.

Questo tipo di **soluzione non è nuova per il settore**. Ad esempio, nel 2006 **Sony** ha rilasciato il sistema **PlayStation** 3 che utilizzava una nuova architettura e non era compatibile con il precedente sistema PlayStation 2 a livello hardware. Lo hanno risolto aggiungendo il chip **PS2** alle prime versioni delle console **PS3** che consentivano di eseguire giochi PS2.

Un altro esempio è un processo di modifica della CPU nei computer **Apple Macintosh** da **PowerPC** a **Intel**. Hanno fornito un simulatore che consente di eseguire l'applicazione PowerPC sui nuovi computer basati su Intel per un periodo di tempo limitato fino a quando la maggior parte delle applicazioni è migrata alla nuova piattaforma.

Come può essere implementata tecnicamente la retrocompatibilità? Una delle soluzioni ovvie e dirette è fornire due implementazioni: una supporta javax e un'altra supporta jakarta. Le implementazioni Javax esistono già come parte di Java EE 8. Le implementazioni Jakarta possono essere create biforcando implementazioni javax e apportando le modifiche necessarie.

Un altro modo consiste nell'applicare patch ai binari dell'applicazione in fase di esecuzione o di compilazione. La soluzione runtime può essere realizzata utilizzando JavaAgent e il tempo di compilazione tramite strumenti e plug-in di compilazione.

Per semplificare la migrazione al pacchetto jakarta a livello di origine, la comunità potrebbe pensare di creare plug-in IDE sostituendo javax con jakarta in modo intelligente con un clic.

Sto lavorando in diversi comitati di Jakarta EE e vedo l'intenzione dei partecipanti di rimanere impegnati nel lavoro di Jakarta EE, finire Jakarta EE 8 e spingerlo in avanti per fornire una piattaforma aziendale robusta, compatibile e innovativa.