**Métodos**

* **setInitialSize:** se utiliza para configurar el tamaño inicial del pool de conexiones. El cual siempre que creemos un pool por primera vez se establecerán y mantendrán 3 conexiones listas para su uso.
* **setMinIdle:** se utiliza para configurar el número mínimo de conexiones inactivas que el pool debe mantener. Se configura para que el pool siempre tenga al menos 3 conexiones inactivas disponibles. Si alguna conexión se cierra o se devuelve al pool, el pool mantendrá al menos 3 de ellas inactivas para futuros usos.
* **setMaxIdle:** se utiliza para configurar el número máximo de conexiones inactivas que el pool debe mantener. Se configura para que el pool pueda tener hasta un máximo de 8 conexiones inactivas.

**Contextos**

* **@Dependent**

Va por defecto e indica que el ciclo de vida del componente depende del ciclo de vida del que lo inyecta. Por lo que cada vez que se inyecta, lo hace en uno más largo, como @RequestScoped o @SessionScoped

* **@RequestScoped**

Crea un bean nuevo para cada solicitud HTTP. El bean se mantiene vivo hasta que la solicitud HTTP se complete.

* **@SessionScoped**

Crea un bean nuevo para cada sesión del usuario, persistiendo durante toda la sesión del usuario en la app web. Además cada usuario tiene su propia instancia de este bean.

* **@ConversationScoped**

Se utilizan en web apps con CDI para mantener una conversación entre varias solicitudes. Además permite que los beans estén vivos durante una conversación específica entre el usuario y la app.

* **@ApplicationScoped**

Comparte un único bean para toda la aplicación, esto asegura que solo exista una sola instancia para toda la app. Y es útil para mantener la información global que debe ser compartida entre múltiples componentes.

**Anotaciones**

* **@Retention**

Se utiliza para especificar cuánto tiempo debe mantenerse en tiempo de ejecución. La cual toma un parámetro “RetentionPolicy” que posee tres valores posibles:

*.SOURCE:* Está disponible en tiempo de compilación y se descarta en tiempo de ejecución

*.CLASS:* Se conserva en el archivo de clases pero no está disponible en tiempo de ejecución.

*.RUNTIME:* Se conserva en tiempo de ejecución y se puede acceder mediante reflexión.

* **@Target**

Se utiliza para especificar en qué elementos se puede aplicar una anotación personalizada. La cual toma como parámetro “ElementType” que especifica los tipos de elementos que pueden usar la anotación personalizada.

* **@Qualifier**

Se utiliza para calificar y distinguir entre diferentes implementaciones de un mismo tipo de bean cuando se realiza la inyección de dependencias, lo que ayuda a resolver ambigüedades cuando hay varias implementaciones para el mismo tipo de bean. Además estas se definen como anotaciones personalizadas y se aplican a las implementaciones concretas de un tipo bean.