अवस्थापन अपवाद

LazyDynaBean.java में 844 लाइन पर java.lang.InstantiationException होने पर, "□□□□□ □□□□□□□□□□□□□ □□□□□□□□□ □□ □□□□"के साथ, निम्नलिखित चरणों का पालन करें। यह अपवाद आम तौर पर तब होता है जब □□□□□□ □□□□□□□ □□□□□□□□□ लाइब्रेरी का LazyDynaBean एक प्रॉपर्टी के प्रकार का एक उदाहरण बनाने का प्रयास करता है लेकिन असफल होता है क्योंकि प्रकार को इंस्टैंशियेट नहीं किया जा सकता है—सबसे अधिक संभवतः यह एक एब्स्ट्रैक्ट क्लास, एक इंटरफेस है, या एक सार्वजनिक नो-आर्ग्यूमेंट कंस्ट्रक्टर का अभाव है।

समस्या को समझना

LazyDynaBean एक डायनामीक बीन इम्प्लीमेंटेशन है जो प्रॉपर्टी को लेज़ी इंटरनेशनलाइज़ेशन के लिए डिज़ाइन किया गया है जब वे एक्सेस किए जाते हैं। त्रुटि सुझाव देता है कि LazyDynaBean.java के 844 लाइन पर कोड एक प्रॉपर्टी के प्रकार का एक उदाहरण बनाने का प्रयास करता है (उदाहरण के लिए, Class.newInstance() का उपयोग करके), लेकिन ऑपरेशन विफल हो जाता है, जो एक InstantiationException फेंकता है। आम कारणों में शामिल हैं: - प्रॉपर्टी का प्रकार एक एब्स्ट्रैक्ट क्लास या इंटरफेस है (उदाहरण के लिए, java.util.List के बजाय java.util.ArrayList). - प्रकार एक कांक्रेट क्लास है लेकिन एक सार्वजनिक नो-आर्ग्यूमेंट कंस्ट्रक्टर का अभाव है, जो newInstance() द्वारा आवश्यक है।

समस्या को ठीक करने के चरण

1. समस्या वाले प्रॉपर्टी को पहचानें

- **स्टैक ट्रेस का जांच करें**: पूर्ण स्टैक ट्रेस या त्रुटि लॉग्स को इंगित करेगा कि LazyDynaBean अपवाद के समय किस प्रॉपर्टी का उदाहरण बनाने का प्रयास कर रहा है। उदाहरण के लिए, यदि अपवाद एक कॉल के दौरान फेंक दिया जाता है जैसे dynaBean.get("someProperty"), तो "□□□□□□□□□□□□□"दोषी है।
- **त्रुटि संदेश का जांच करें**: यदि पूर्ण त्रुटि संदेश प्रकार को स्पष्ट करता है (उदाहरण के लिए, "□□□□□ □□□□□□□□□□□□□ □□□□□□□□□ □□ □□□□ □□□□.□□□□.□□□□"), तो शामिल प्रकार को नोट करें।

2. प्रॉपर्टी का प्रकार निर्धारित करें

- DynaClass **कॉन्फ़िगरेशन का जांच करें**: LazyDynaBean एक DynaClass (अक्सर एक LazyDynaClass) पर निर्भर करता है ताकि इसकी प्रॉपर्टी और उनके प्रकार को परिभाषित किया जा सके। जांच करें कि प्रॉपर्टी कैसे परिभाषित हैं:
  - अगर आपने एक LazyDynaClass को स्पष्ट रूप से बनाया है, तो प्रॉपर्टी जोड़ने वाले कोड को देखें, जैसे dynaClass.add("propertyName", PropertyType.class).
  - अगर LazyDynaBean को एक पूर्व-परिभाषित DynaClass के बिना बनाया गया है (उदाहरण के लिए, new LazyDynaBean()), तो प्रॉपर्टी को डायनामीक रूप से जोड़ा जाता है, और प्रकार को पहली सेट की गई मान से या एक समस्या वाले प्रकार पर डिफ़ॉल्ट हो सकता है।
- **डिबगिंग टिप**: लॉगिंग जोड़ें या डिबगर का उपयोग करें ताकि dynaClass.getDynaProperty("propertyName").getType() द्वारा लौटाए गए प्रकार को प्रिंट करें।

3. प्रॉपर्टी प्रकार को इंस्टैंशियेट करने योग्य बनाएं

- **एक कांक्रेट क्लास का उपयोग करें**: अगर प्रकार एक एब्स्ट्रैक्ट क्लास या इंटरफेस है (उदाहरण के लिए, List, Map, या एक कस्टम इंटरफेस MyInterface), तो इसे एक सार्वजनिक नो-आर्ग्यूमेंट कंस्ट्रक्टर वाले कांक्रेट इम्प्लीमेंटेशन से बदलें:

- `List` के लिए, `ArrayList.class` का उपयोग करें `List.class` के बजाय।
- `Map` के लिए, `HashMap.class` का उपयोग करें `Map.class` के बजाय।
- एक कस्टम इंटरफेस या एब्स्ट्रैक्ट क्लास के लिए, एक कांक्रेट सबक्लास का चयन करें (उदाहरण के लिए, `MyInterface` को इम्प्लीमेंट करने वाला `MyConcreteClass`).

 उदाहरण:

```
LazyDynaClass dynaClass = new LazyDynaClass();
dynaClass.add("myList", ArrayList.class); //
LazyDynaBean dynaBean = new LazyDynaBean(dynaClass);
```

4. कॉन्फ़िगरेशन को समायोजित करें

 प्रॉपर्टी को पूर्व-परिभाषित करें: अगर आप `DynaClass` को नियंत्रित करते हैं, तो कांक्रेट प्रकारों के साथ प्रॉपर्टी को स्पष्ट रूप से परिभाषित करें:

```
LazyDynaClass dynaClass = new LazyDynaClass();
dynaClass.add("myProperty", MyConcreteClass.class);
LazyDynaBean dynaBean = new LazyDynaBean(dynaClass);
```

 प्रारंभिक मान सेट करें: या तो, प्रॉपर्टी को एक्सेस करने से पहले एक कांक्रेट क्लास का एक प्रारंभिक उदाहरण सेट करें, जिससे `LazyDynaBean` को इसे इंस्टैंशियेट करने का प्रयास करने से रोकें:

```
LazyDynaBean dynaBean = new LazyDynaBean();
dynaBean.set("myProperty", new ArrayList<>()); //
Object value = dynaBean.get("myProperty"); //
```

5. डायनामीक प्रॉपर्टी निर्माण को संभालें

 अगर प्रॉपर्टी को डायनामीक रूप से बनाया जाता है (जिसे `LazyDynaBean` के साथ आम तौर पर होता है), तो सुनिश्चित करें कि प्रॉपर्टी के लिए सेट की गई पहली मान एक कांक्रेट क्लास का उदाहरण है। यह प्रकार को सही तरह से सेट करता है:

```
LazyDynaBean dynaBean = new LazyDynaBean();
dynaBean.set("myList", new ArrayList<>()); //       ArrayList
```

 बिना पहले सेट किए अनिर्धारित प्रॉपर्टी को एक्सेस करने से बचें, क्योंकि `LazyDynaBean` एक डिफ़ॉल्ट प्रकार को इंस्टैंशियेट करने का प्रयास कर सकता है जो समस्या हो सकता है।

6. कंस्ट्रक्टर के एक्सेस को सत्यापित करें

 सुनिश्चित करें कि कांक्रेट क्लास में एक सार्वजनिक नो-आर्ग्यूमेंट कंस्ट्रक्टर है। उदाहरण के लिए:

```
public class MyConcreteClass {
    public MyConcreteClass() {
        //       -
```

```
        }
    }
```

- अगर क्लास में यह नहीं है, तो इसे जोड़ें या इसे पूरा करने वाले किसी अन्य क्लास का उपयोग करें।

उदाहरण समाधान

अगर त्रुटि इसलिए होती है क्योंकि एक प्रॉपर्टी "items" को java.util.List के रूप में टाइप किया गया है, एक इंटरफेस: - **समस्या वाले कोड**: java LazyDynaClass dynaClass = new LazyDynaClass(); dynaClass.add("items", List.class); // , LazyDynaBean dynaBean = new LazyDynaBean(dynaClass); dynaBean.get("items"); // InstantiationException - **सुधार किया गया कोड**: java LazyDynaClass dynaClass = new LazyDynaClass(); dynaClass.add("items", ArrayList.class); // LazyDynaBean dynaBean = new LazyDynaBean(dynaClass); dynaBean.get("items"); // ArrayList

अतिरिक्त विचार

- **कस्टम प्रकार**: अगर प्रॉपर्टी का प्रकार एक कस्टम एब्स्ट्रैक्ट क्लास या इंटरफेस है, तो अपने एप्लिकेशन के लिए एक कांक्रेट सबक्लास प्रदान करें।
- **लाइब्रेरी संस्करण**: सुनिश्चित करें कि आप एक संगत संस्करण का उपयोग कर रहे हैं □□□□□□ □□□□□□□ □□□□□□□□□, हालांकि यह आम तौर पर उपयोग संबंधी है, नहीं तो एक बग.
- **वैकल्पिक दृष्टिकोण**: अगर प्रकारों को समायोजित करना असंभव है, तो सभी प्रॉपर्टी को कांक्रेट उदाहरणों के साथ प्रारंभिक रूप से सेट करने का विचार करें, या WrapDynaBean का उपयोग करें ताकि बीन को इंस्टैंशियेट करने के प्रयासों के बिना लपेटा जा सके।

अंतिम समाधान

LazyDynaBean.java में 844 लाइन पर java.lang.InstantiationException को हल करने के लिए: 1. स्टैक ट्रेस या लॉग्स के माध्यम से त्रुटि को उत्पन्न करने वाले प्रॉपर्टी को पहचानें। 2. DynaClass में या इसके उपयोग से अनुमानित प्रकार को जांचें। 3. किसी भी एब्स्ट्रैक्ट क्लास या इंटरफेस को एक कांक्रेट क्लास (उदाहरण के लिए, ArrayList के लिए List) से बदलें जो एक सार्वजनिक नो-आर्ग्यूमेंट कंस्ट्रक्टर रखता है। 4. DynaClass परिभाषा को अपडेट करें या एक प्रारंभिक कांक्रेट उदाहरण सेट करें ताकि स्वचालित इंस्टैंशियेशन से बचा जा सके।

त्रुटि संदेश से विशेष प्रॉपर्टी नाम या प्रकार के बिना, सुनिश्चित करें कि LazyDynaBean का उपयोग करते समय सभी प्रॉपर्टी प्रकार कांक्रेट और इंस्टैंशियेट करने योग्य हैं। यह अपवाद को समाप्त कर देगा।

मैं समझता हूँ कि आप अपने फिल्टर क्लास में init(FilterConfig filterConfig) विधि के दौरान java.lang.InstantiationException का सामना कर रहे हैं, और आप इसे ठीक करने के लिए एक तरीका ढूंढ रहे हैं। यह त्रुटि तब होती है जब कोड एक उदाहरण बनाने का प्रयास करता है जो एक इंस्टैंशियेट नहीं किया जा सकता, जैसे एक एब्स्ट्रैक्ट क्लास या इंटरफेस, जो आम तौर पर LazyDynaBean का उपयोग करने के साथ 844 लाइन पर जुड़ा हुआ है। क्योंकि आपने केवल एक फिल्टर क्लास के साथ एक init विधि का उल्लेख किया है, तो इसे चरण-दर-चरण हल करने के लिए:

समस्या को समझना

एक जावा वेब एप्लिकेशन में, एक फिल्टर की `init` विधि को सर्वर कंटेनर द्वारा एक `FilterConfig` ऑब्जेक्ट का उपयोग करके फिल्टर को इंटरनेशनलाइज़ करने के लिए कॉल किया जाता है। त्रुटि सुझाव देता है कि इस इंटरनेशनलाइज़ेशन के दौरान कुछ (संभावित रूप से `LazyDynaBean`) एक प्रॉपर्टी प्रकार का एक उदाहरण बनाने का प्रयास कर रहा है जो एक कांक्रेट क्लास नहीं है या एक सार्वजनिक नो-आर्ग्यूमेंट कंस्ट्रक्टर का अभाव है। क्योंकि आप `LazyDynaBean` का उपयोग कर रहे हैं (त्रुटि संदेश से संकेत मिलता है), यह संभवतः फिल्टर कंफ़िगरेशन डेटा को स्टोर करने या इंटरनेशनलाइज़ेशन पैरामीटर को प्रोसेस करने के लिए डायनामीक रूप से प्रॉपर्टी को संभालने के लिए उपयोग किया जा रहा है।

समस्या को ठीक करने के चरण

1. **अपने `init` विधि का जांच करें** अपने `init(FilterConfig filterConfig)` विधि के कोड को देखें। आप संभवतः एक `LazyDynaBean` को फिल्टर कंफ़िगरेशन डेटा को स्टोर करने या प्रारंभिक पैरामीटर को प्रोसेस करने के लिए बनाते हैं। उदाहरण के लिए, आपके कोड का यह रूप हो सकता है:

```
import org.apache.commons.beanutils.LazyDynaBean;
import javax.servlet.*;

public class MyFilter implements Filter {
    private LazyDynaBean configBean;

    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
        configBean = new LazyDynaBean();
        Enumeration<String> initParams = filterConfig.getInitParameterNames();
        while (initParams.hasMoreElements()) {
            String paramName = initParams.nextElement();
            String paramValue = filterConfig.getInitParameter(paramName);
            configBean.set(paramName, paramValue);
        }
        //
        Object someProperty = configBean.get("someProperty");
    }

    @Override
    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
            throws IOException, ServletException {
        chain.doFilter(request, response);
    }

    @Override
```

```
    public void destroy() {}
}
```

इस उदाहरण में, अगर `"someProperty"` पहले से मान के साथ सेट नहीं किया गया है और इसका प्रकार एक एब्स्ट्रैक्ट है (उदाहरण के लिए, `List` के बजाय `ArrayList`), तो `LazyDynaBean` इसे इंस्टैंशियेट करने का प्रयास करेगा और विफल हो जाएगा, जिससे `InstantiationException` फेंकता है।

2. **समस्या वाले प्रॉपर्टी को पहचानें** क्योंकि त्रुटि `LazyDynaBean.java` में 844 लाइन पर होती है, यह संभवत: एक `get` या `set` ऑपरेशन के साथ `LazyDynaBean` से जुड़ा हुआ है। इसे पहचानने के लिए:

   - प्रत्येक `configBean.get()` या `configBean.set()` कॉल से पहले लॉगिंग या प्रिंट स्टेटमेंट जोड़ें ताकि त्रुटि को ट्रिगर करने वाला प्रॉपर्टी कौन है।
   - उदाहरण:

     ```
     System.out.println("               : someProperty");
     Object someProperty = configBean.get("someProperty");
     ```

3. **कांक्रेट प्रकार या प्रारंभिक मान सुनिश्चित करें** `LazyDynaBean` प्रॉपर्टी को लेज़ी इंटरनेशनलाइज़ेशन करता है, लेकिन अगर आप एक प्रॉपर्टी को पहले से सेट नहीं करते हैं, तो यह इसका प्रकार इंस्टैंशियेट करने का प्रयास करता है। अगर वह प्रकार एक एब्स्ट्रैक्ट या इंटरफेस है (उदाहरण के लिए, `List`, `Map`), तो यह एक `InstantiationException` फेंकता है। इसे ठीक करने के लिए:

   - **एक प्रारंभिक मान सेट करें**: एक कांक्रेट उदाहरण से पहले प्रॉपर्टी को सेट करें।

     ```
     configBean.set("someProperty", new ArrayList<String>()); //
     Object someProperty = configBean.get("someProperty");    //
     ```

   - **एक कांक्रेट प्रकार परिभाषित करें**: अगर आप प्रॉपर्टी प्रकार परिभाषित करते हैं, तो कांक्रेट क्लास का उपयोग करें।

     ```
     configBean.setType("someProperty", ArrayList.class); //    `List.class`
     ```

4. **कंस्ट्रक्टर की सत्यापना करें** अगर आप एक कस्टम क्लास (उदाहरण के लिए, `MyCustomClass`) के साथ एक प्रॉपर्टी सेट करते हैं, तो सुनिश्चित करें कि इसमें एक सार्वजनिक नो-आर्ग्यूमेंट कंस्ट्रक्टर है:

   ```
   public class MyCustomClass {
       public MyCustomClass() {} // `LazyDynaBean`
   }
   ```

   फिर इसे उपयोग करें:

   ```
   configBean.set("someProperty", new MyCustomClass());
   ```

5. **साधारण बनाएं यदि संभव हो** अगर आप केवल `LazyDynaBean` का उपयोग `FilterConfig` पैरामीटर को स्टोर करने के लिए कर रहे हैं, तो इसे पूरी तरह से छोड़ने का विचार करें और एक `Map` या सीधे `FilterConfig` एक्सेस का उपयोग करें:

   ```
   public class MyFilter implements Filter {
       private Map<String, String> configMap;

       @Override
   ```

```
    public void init(FilterConfig filterConfig) throws ServletException {
        configMap = new HashMap<>();
        Enumeration<String> initParams = filterConfig.getInitParameterNames();
        while (initParams.hasMoreElements()) {
            String paramName = initParams.nextElement();
            configMap.put(paramName, filterConfig.getInitParameter(paramName));
        }
    }
    //      ...
}
```

यह LazyDynaBean की जटिलताओं को पूरी तरह से बर्खास्त कर देता है।

अंतिम समाधान

LazyDynaBean की आवश्यकता होने पर, एक सुधारित फिल्टर का उदाहरण:

```
import org.apache.commons.beanutils.LazyDynaBean;
import javax.servlet.*;
import java.util.ArrayList;
import java.util.Enumeration;

public class MyFilter implements Filter {
    private LazyDynaBean configBean;

    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
        configBean = new LazyDynaBean();
        Enumeration<String> initParams = filterConfig.getInitParameterNames();
        while (initParams.hasMoreElements()) {
            String paramName = initParams.nextElement();
            String paramValue = filterConfig.getInitParameter(paramName);
            configBean.set(paramName, paramValue); //
        }
        //
        if (configBean.get("someProperty") == null) {
            configBean.set("someProperty", new ArrayList<String>());
        }
        Object someProperty = configBean.get("someProperty"); //
    }
```

```
    @Override
    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
            throws IOException, ServletException {
        chain.doFilter(request, response);
    }

    @Override
    public void destroy() {}
}
```

समाधान का परीक्षण

- अपडेटेड फिल्टर को सर्वर कंटेनर (उदाहरण के लिए, टोमकैट) में डिप्लॉय करें।
- लॉग्स को जांचें ताकि `InstantiationException` अब नहीं होती।
- अगर यह जारी है, तो पूर्ण स्टैक ट्रेस या अधिक कोड विवरण के साथ और मदद के लिए साझा करें।

यह दृष्टिकोण सुनिश्चित करता है कि `LazyDynaBean` द्वारा एक्सेस किए गए सभी प्रॉपर्टी पहले से कांक्रेट उदाहरणों के साथ सेट किए गए हैं या टाल दिए गए हैं, जिससे फिल्टर की `init` विधि में `InstantiationException` समाप्त हो जाता है।

मैं समझता हूँ कि आपने एक त्रुटि संदेश में `LazyDynaBean` का उल्लेख किया है (विशेष रूप से एक `java.lang.InstantiationException` `LazyDynaBean.java:844` पर), जबकि आपने अपने कोड में `LazyDynaBean` का उल्लेख या उपयोग नहीं किया है। यह स्थिति तब होती है जब आपके प्रोजेक्ट में एक लाइब्रेरी या फ्रेमवर्क `LazyDynaBean` को अप्रत्यक्ष रूप से उपयोग करती है। इसे चरण-दर-चरण हल करने के लिए:

`LazyDynaBean` क्यों शामिल है?

क्योंकि आपने स्पष्ट रूप से `LazyDynaBean` का उल्लेख नहीं किया है, यह संभवतः आपके प्रोजेक्ट में एक निर्भरता द्वारा बुलाया जा रहा है, जैसे: - □□□□□□□ □□□□□□□□ □□□□□□□□□□: यह लाइब्रेरी `LazyDynaBean` को आंतरिक रूप से डायनामीक प्रॉपर्टी एक्सेस या बीन प्रबंधन के लिए उपयोग करती है। - □□□□□□□ **फ्रेमवर्क**: अगर आपका कोड स्प्रिंग के साथ एकीकृत है (उदाहरण के लिए, एक फिल्टर या बीन कॉन्फ़िगरेशन में), तो यह □□□□□□□□□□ या समान उपयोगीयों का उपयोग कर सकता है जो `LazyDynaBean` से जुड़े होते हैं। - **अन्य फ्रेमवर्क**: किसी भी टूल जो डायनामीक प्रॉपर्टी, कॉन्फ़िगरेशन, या बीन इंस्टैंशियेशन के साथ निपटता है, यह हो सकता है।

`InstantiationException` सुझाव देता है कि `LazyDynaBean` एक प्रकार का एक उदाहरण बनाने का प्रयास कर रहा है लेकिन असफल हो रहा है, संभवतः क्योंकि यह एक एब्स्ट्रैक्ट क्लास, एक इंटरफेस, या एक सार्वजनिक नो-आर्ग्यूमेंट कंस्ट्रक्टर का अभाव के साथ मिलता है।

समस्या को ठीक करने के लिए

यहाँ एक संरचित दृष्टिकोण है:

1. स्टैक ट्रेस का जांच करें

- पूर्ण `InstantiationException` के स्टैक ट्रेस को देखें। यह `LazyDynaBean.java:844` तक के कॉल की श्रृंखला को दिखाएगा।
- कोड में लाइब्रेरी या फ्रेमवर्क को पहचानें जो इस कॉल को ट्रिगर करती है। उदाहरण के लिए, आप `org.apache.commons.beanutils` या `org.springframework.beans` के संदर्भ देख सकते हैं।

2. अपने कोड और निर्भरताओं का जांच करें

- अपने फिल्टर (या त्रुटि के स्थान पर क्लास) के निर्भरताओं को जांचें। अगर यह एक सर्वलेट फिल्टर है, तो देखें:
  - `init` विधि.
  - किसी भी प्रॉपर्टी या बीन.
  - प्रोजेक्ट में आयात किए गए लाइब्रेरी (उदाहरण के लिए, □□□□□/□□□□□□ के माध्यम से).
- आम लाइब्रेरी को संदेह में लेना चाहिए:
  - `commons-beanutils` (डायनामीक प्रॉपर्टी प्रबंधन के लिए उपयोग किया जाता है).
  - स्प्रिंग या अन्य फ्रेमवर्क जो बीन प्रबंधन करते हैं।

3. कॉन्फ़िगरेशन का जांच करें

- अगर आपका फिल्टर एक □□□ (उदाहरण के लिए, एक `web.xml` या स्प्रिंग कॉन्फ़िगरेशन फ़ाइल) के माध्यम से कॉन्फ़िगर किया गया है, तो सुनिश्चित करें कि सभी संदर्भित ऑब्जेक्ट सही तरह से परिभाषित हैं।
- उदाहरण के लिए, अगर एक प्रॉपर्टी डायनामीक रूप से सेट किया जाता है:

```
<bean id="myFilter" class="com.example.MyFilter">
    <property name="someProperty" ref="someBean"/>
</bean>
```

सुनिश्चित करें कि `someBean` एक कांक्रेट क्लास है, जैसे:

```
<bean id="someBean" class="com.example.ConcreteClass"/>
```

4. कांक्रेट प्रकार सुनिश्चित करें

- अपवाद आम तौर पर तब होता है जब एक लाइब्रेरी एक प्रकार को इंस्टैंशियेट करने की उम्मीद करता है लेकिन एक इंटरफेस या एब्स्ट्रैक्ट क्लास (उदाहरण के लिए, `List` के बजाय `ArrayList`) को मिलता है।
- अगर आप प्रॉपर्टी परिभाषित करते हैं, तो सुनिश्चित करें कि वे सार्वजनिक नो-आर्ग्यूमेंट कंस्ट्रक्टर वाले कांक्रेट इम्प्लीमेंटेशन का उपयोग करते हैं:

```
private List<String> myList = new ArrayList<>(); //
private List<String> myList; //
```

5. समस्या का डिबग करें

- अपने फिल्टर की `init` विधि (या त्रुटि के स्थान पर) में लॉगिंग जोड़ें या डिबगर का उपयोग करें ताकि त्रुटि को ट्रिगर करने वाले ठीक से लाइन को पहचानें।
- उदाहरण:

```
public class MyFilter implements Filter {
    private List<String> myList;

    public void setMyList(List<String> myList) {
        this.myList = myList;
    }

    @Override
    public void init(FilterConfig config) throws ServletException {
        System.out.println("                ...");
        if (myList == null) {
            myList = new ArrayList<>(); //
        }
    }
}
```

6. मैन्युअल इंटरनेशनलाइज़ेशन (अगर आवश्यक हो)

- अगर लाइब्रेरी डायनामीक रूप से प्रॉपर्टी सेट करती है और विफल होती है, तो उन्हें स्वयं इंटरनेशनलाइज़ करने का विचार करें ताकि समस्या से बचा जा सके:

```
public class MyFilter implements Filter {
    private SomeClass myObject = new SomeClass(); //

    @Override
    public void init(FilterConfig config) throws ServletException {
        //
    }
}
```

उदाहरण स्थिति

अगर आप स्प्रिंग के `GenericFilterBean` का उपयोग कर रहे हैं और आपके पास है:

```
<bean id="myFilter" class="com.example.MyFilter">
    <property name="myList" ref="listBean"/>
```

```
</bean>
<bean id="listBean" class="java.util.List"/> <!--    :     -->
```

सुधार होगा:

```
<bean id="listBean" class="java.util.ArrayList"/> <!--        -->
```

या कोड में:

```
public class MyFilter extends GenericFilterBean {
    private List<String> myList = new ArrayList<>(); //

    public void setMyList(List<String> myList) {
        this.myList = myList;
    }
}
```

अंतिम चरण

- **स्रोत को पहचानें**: `LazyDynaBean` को बुलाने वाले लाइब्रेरी या कॉन्फ़िगरेशन को पहचानें।
- **कॉन्फ़िगरेशन को ठीक करें**: सुनिश्चित करें कि सभी प्रकार कांक्रेट और इंस्टैंशियेट करने योग्य हैं।
- **परीक्षण करें**: अपडेटेड फिल्टर को फिर से डिप्लॉय करें और सुनिश्चित करें कि अपवाद समाप्त हो गया है।

अगर समस्या जारी है, तो अधिक विवरण (उदाहरण के लिए, स्टैक ट्रेस, आपका फिल्टर कोड, या निर्भरताएँ) साझा करें, और मैं एक अधिक विशिष्ट समाधान प्रदान कर सकता हूँ। यह दृष्टिकोण आपको `LazyDynaBean` को सीधे संभालने के बिना `InstantiationException` को समाप्त करने में मदद करेगा।