Faculdades Integradas Camões

Cristian Pablo Aparecido da Silva

Gabriel Venâncio de Brito

Curitiba-PR-Brasil

1-Boas Práticas para desenvolvimento JAVA

Um bom programador não é aquele que apenas escreve um código e sim quem escreve o programa, isso significa escrever as unidades de código que são independentes o suficiente para serem reutilizadas de várias maneiras, e ainda sim permanecem robustas.

2-Práticas recomendadas para testes unitários

O teste unitário não é sobre encontrar bugs. Os testes unitários não são uma maneira eficaz de encontrar erros ou detectar regressões. Testes unitários, por definição, examinam cada unidade do seu código separadamente. Mas quando sua aplicação é executada de verdade, todas essas unidades precisam trabalhar juntas, e o todo é mais complexo e sutil do que a soma de suas partes testadas independentemente. A comprovação de que os componentes X e Y funcionam de forma independente não prova que eles sejam compatíveis entre si ou configurados corretamente.Então, se você está tentando encontrar bugs, é muito mais eficaz executar todo o aplicativo juntos, já que ele será executado na produção, assim como você faz naturalmente ao testar manualmente. Se você automatizar esse tipo de teste para detectar rupturas quando elas acontecerem no futuro, ele será chamado de teste de integração e, normalmente, usa técnicas e tecnologias diferentes das do teste de unidade.

E importante também testar apenas uma unidade de código por vez.

Acaba sendo muito fácil escrever testes unitários que inválidos que adicionam muito pouco valor a um projeto enquanto acabam aumentando absurdamente o custo de alteração do código. Uma nova abordagem usa classes internas estáticas para cada novo cenário excepcional.

3-Práticas recomendadas de codificação

Seguindo as melhores práticas de Java, concentram-se nas considerações a serem feitas ao tocar no teclado e começar a digitar o programa real. Eles são principalmente úteis para desenvolvedores em todos os níveis.

Tambem podendo gerar hash de senha segura em produção pois a senha fornecida pelo usuário geralmente é muito fraca e fácil de adivinhar.

Existem muitos algoritmos de hash em java, que podem ser realmente eficazes para a segurança de senha para seus aplicativos e seus usuários, como por exemplo a linha de codigo abaixo.

```
MessageDigest algorithm = MessageDigest.getInstance("MD5");

byte messageDigest[] = algorithm.digest("senha".getBytes("UTF-8"));
```

4-Use array em vez de Vector.elementAt () dentro de qualquer loop

Vector é outra implementação legada da interface List fornecida com o pacote java. É quase semelhante ao ArrayList, espera-se que seja sincronizado também. Tem as suas vantagens e desvantagens, e. O vetor não precisa de sincronização adicional enquanto é acessado de vários segmentos, mas degrada seu desempenho pelo mesmo motivo.

O vetor não precisa de sincronização adicional enquanto é acessado de vários segmentos, mas degrada seu desempenho pelo mesmo motivo. Meça o impacto.

Ex:

Método 1) Usando o método elementAt ()

Desta forma, vamos usar a prática normal para acessar elementos, ou seja, usando elementsAt ().

```
int size = v.size();

for(int i=size; i>0; i--)

{

String str = v.elementAt(size-i);
```

}

Método 2) Usando o array from toArray ()

int size = v.size();

String vArr[] = (String[]) v.toArray();

for(int i=0; i < size(); i--)

5-Sempre use length () em vez de equals () para verificar a string vazia

A melhor maneira de verificar se a string está vazia ou não é usar o método length (). Esse método simplesmente retorna a contagem de caracteres dentro da matriz char que constitui a string. Se a contagem ou comprimento é 0; você pode concluir com segurança que a string está vazia.

EX:

Não fazer isso.

public boolean isEmpty(String str)

{

return str.equals("");

}

E sim fazer isso.

public boolean isEmpty(String str)

{

return str.length()==0;

}

6-Referencias

http://blog.caelum.com.br/guardando-senhas-criptografadas-em-java/

https://www.geeksforgeeks.org/arraylist-array-conversion-java-toarray-methods/

https://www.devmedia.com.br/aprender-java-boas-praticas-para-um-bom-projeto-de-software/29713